How can we trust 3rd party code?

Using Python to understand the trust relationships within the python ecosystem

Nigel Brown

Supply chain attacks

Recent high profile examples

- SolarWinds —Attackers injected a backdoor into a software update of SolarWinds
- Kaseya — Attackers compromised this, infecting it with REvil ransomware
- Atlassian — Atlassian applications were vulnerable to SSO abuse.
- Apple and Microsoft — Security researcher able to hack corporate systems using fake versions of a dependency.
- Mimecast — Hackers were able to compromise the security certificate.
- Codecov — Infected the uploader, injecting malicious code, eavesdropped on Codecov servers and stole customer data.
- British Airways — Magecart supply chain attack disrupted its trading system and leaked sensitive information.

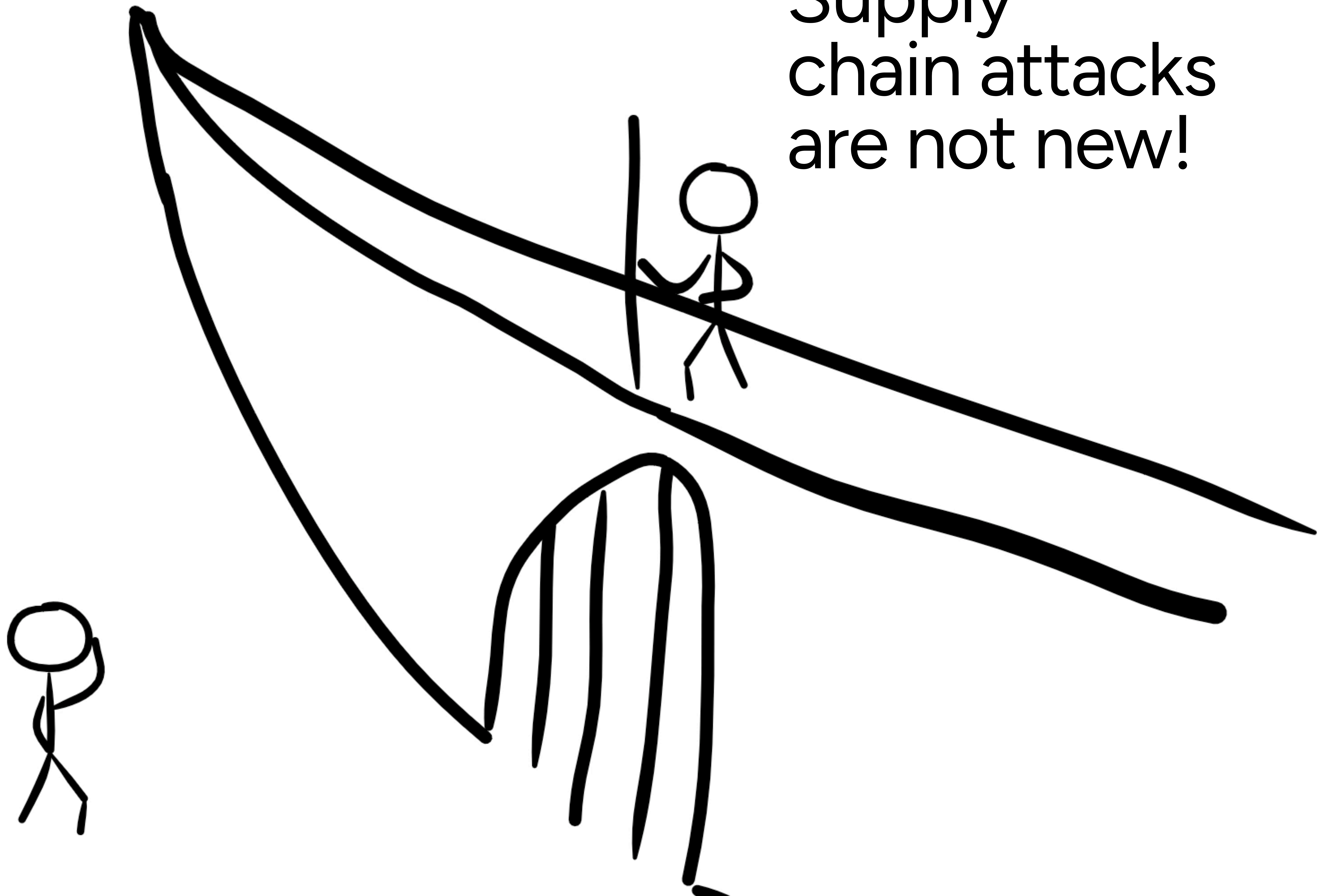
https://www.bluevoyant.com/knowledge-center/supply-chain-attacks-7-examples-and-4-defensive-strategies

Supply chain

Responsibility

- Executive Order 14028
- EU has Cyber Resilience proposal
- Responsibility shifting to the vendor
- Responsibility shifting to you…

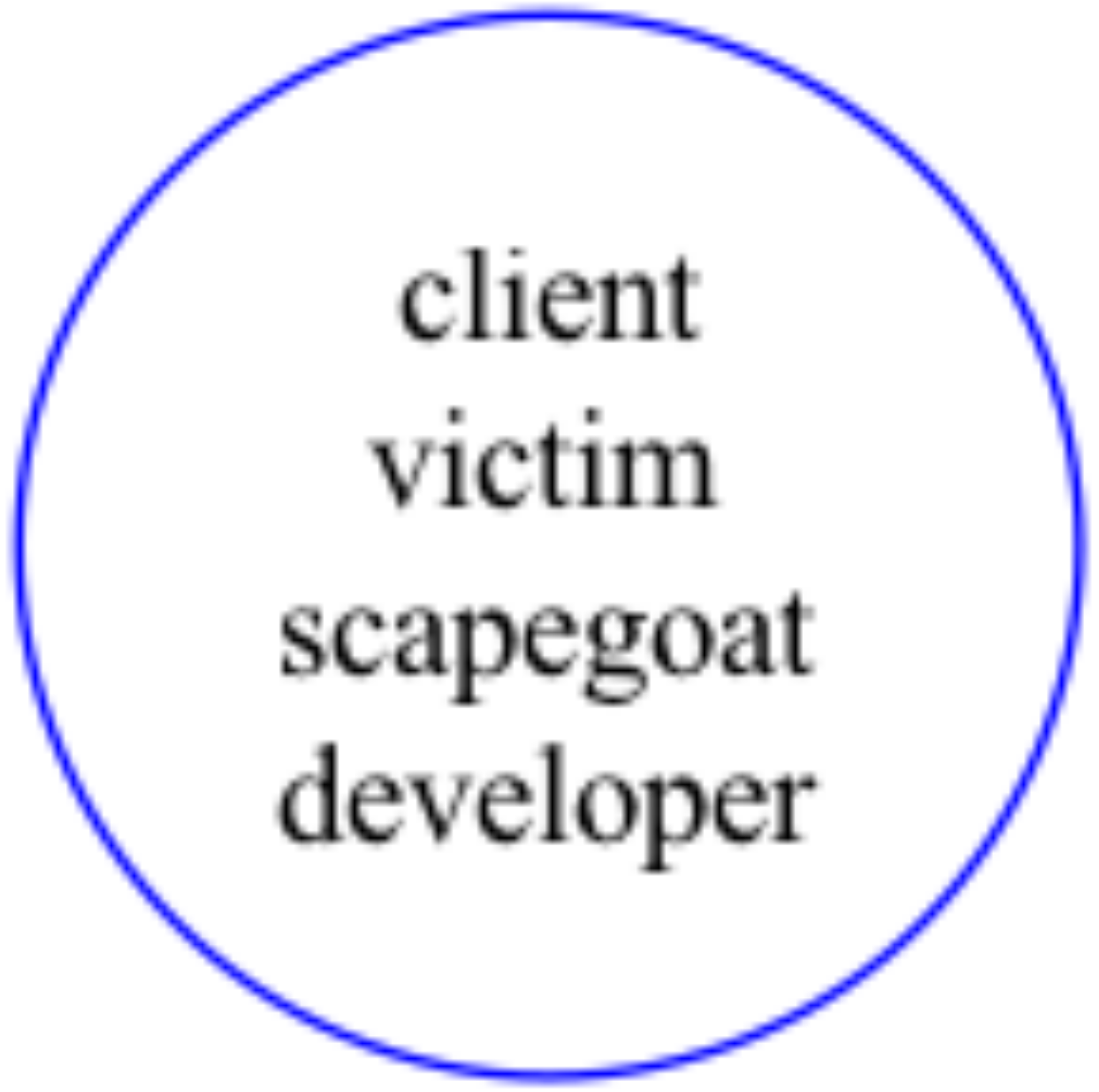
Supply chain attacks are not new!



Web of trust

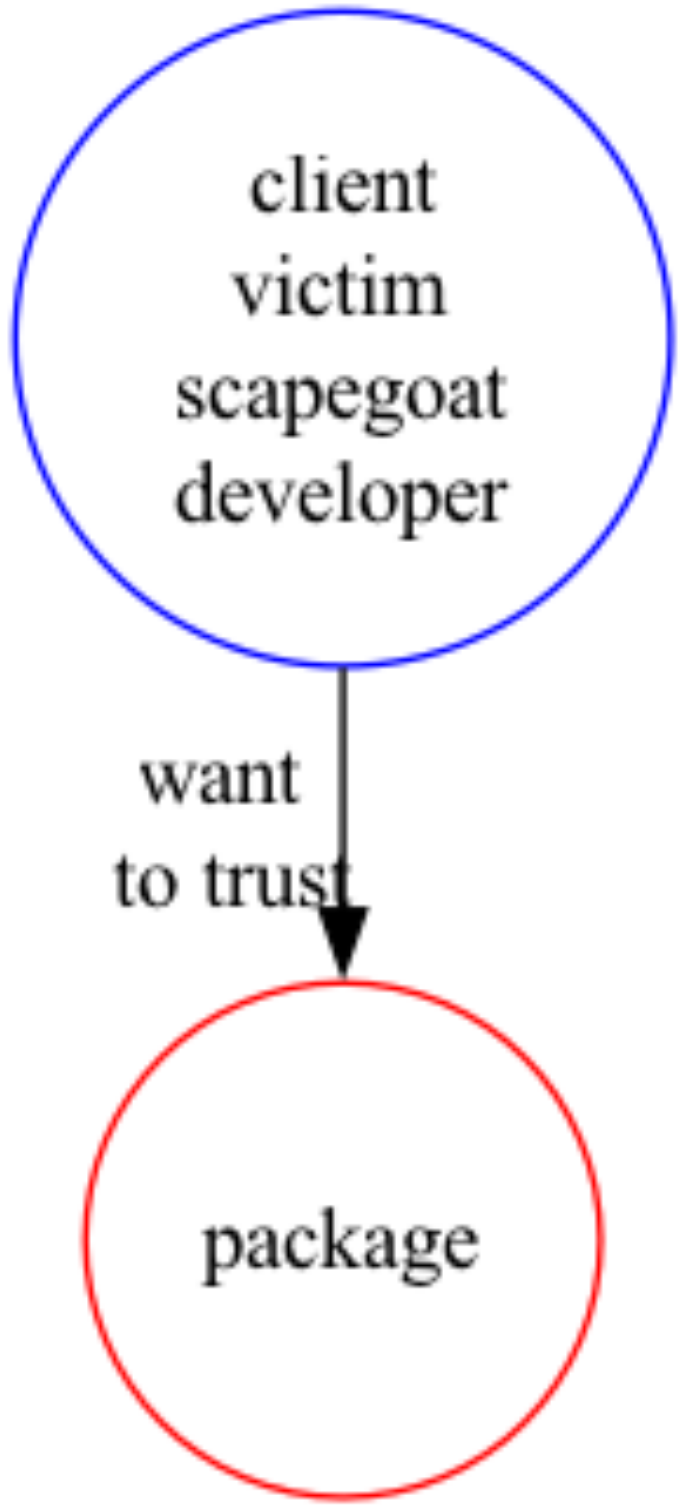
As developers we want to trust 3rd party code

- This is the supply chain
- How can we trust it?...



Web of trust

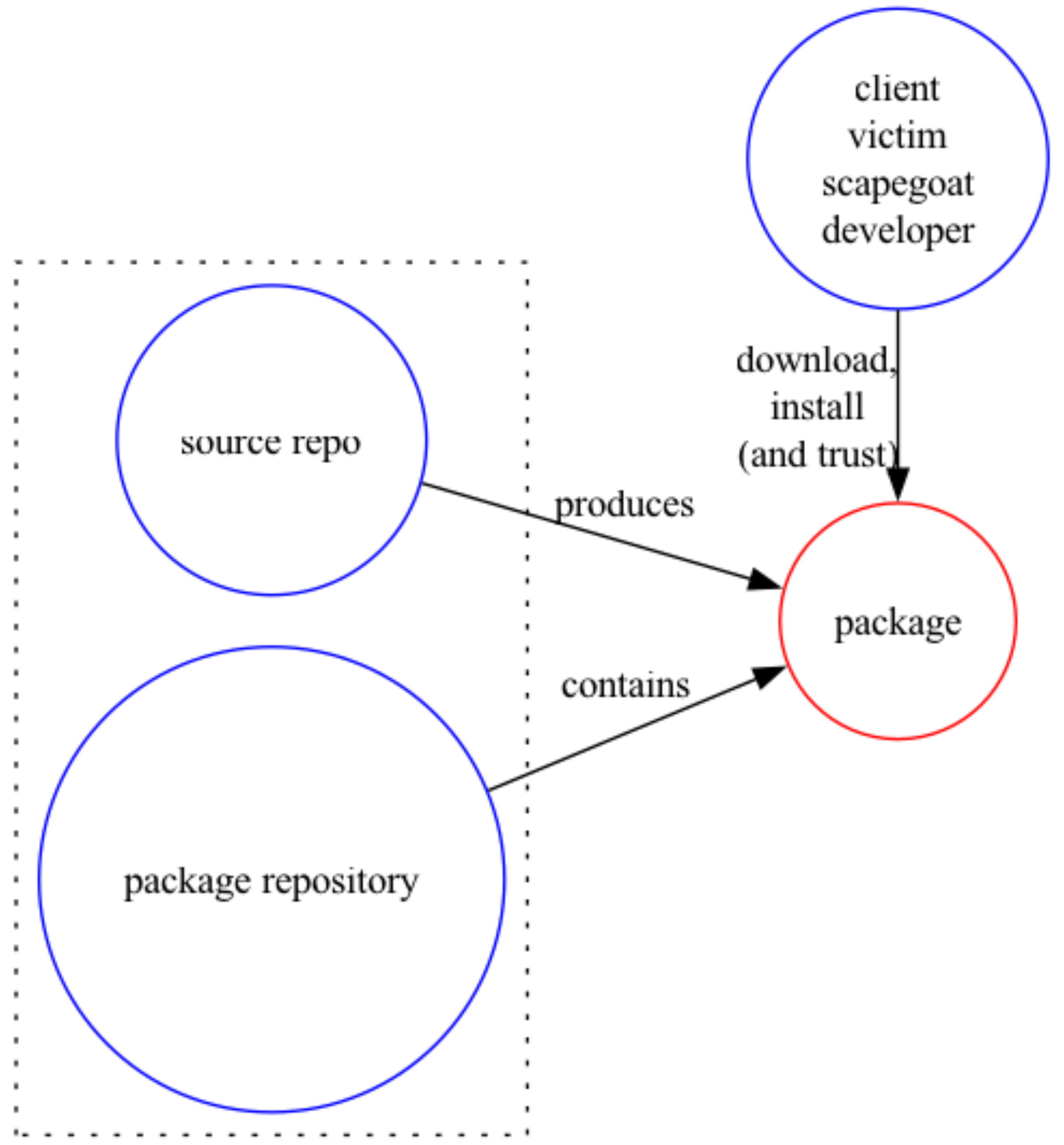
Delivered as some sort of package



Web of trust

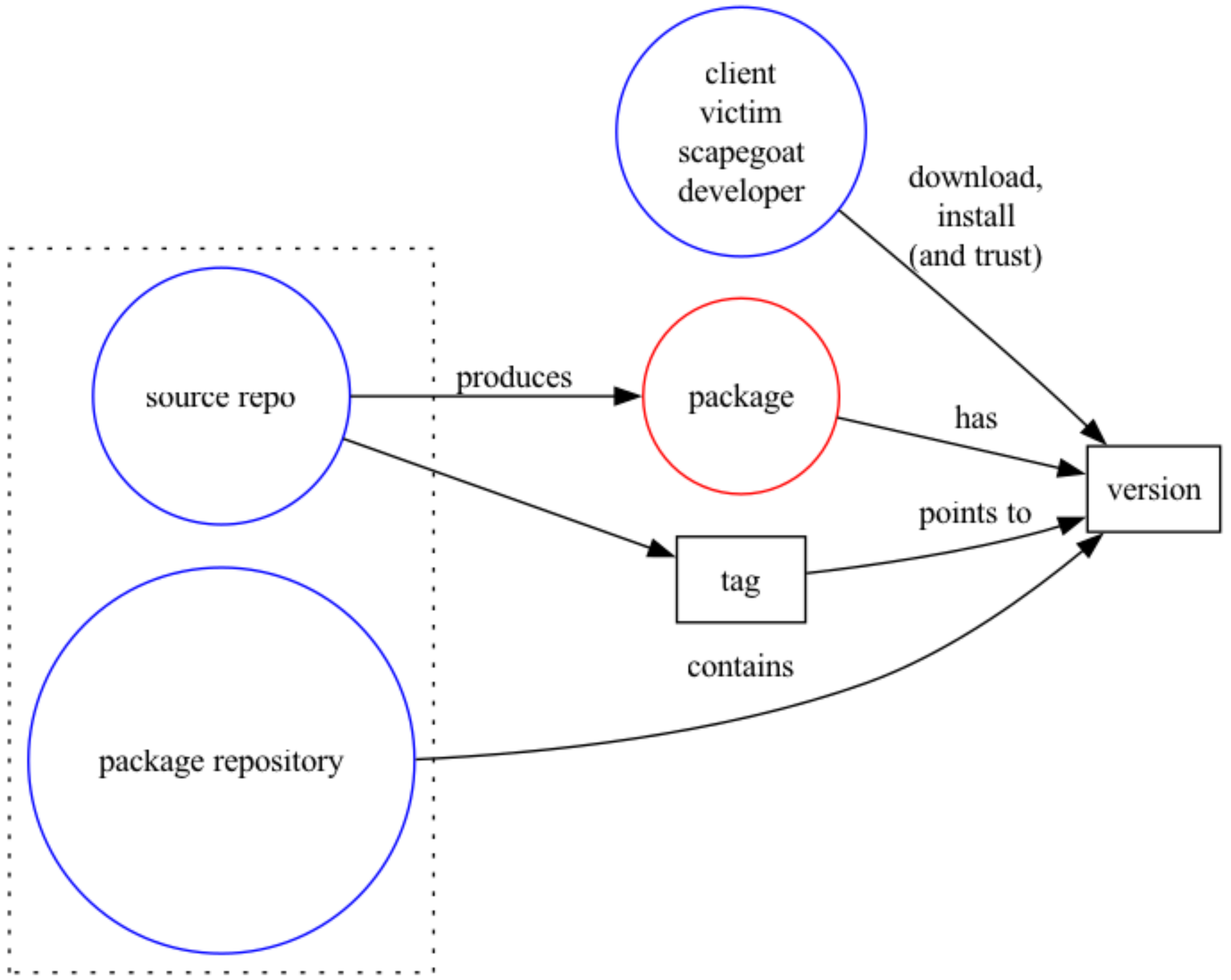
The package and its source live somewhere

- We have a source repository
- And a package repository



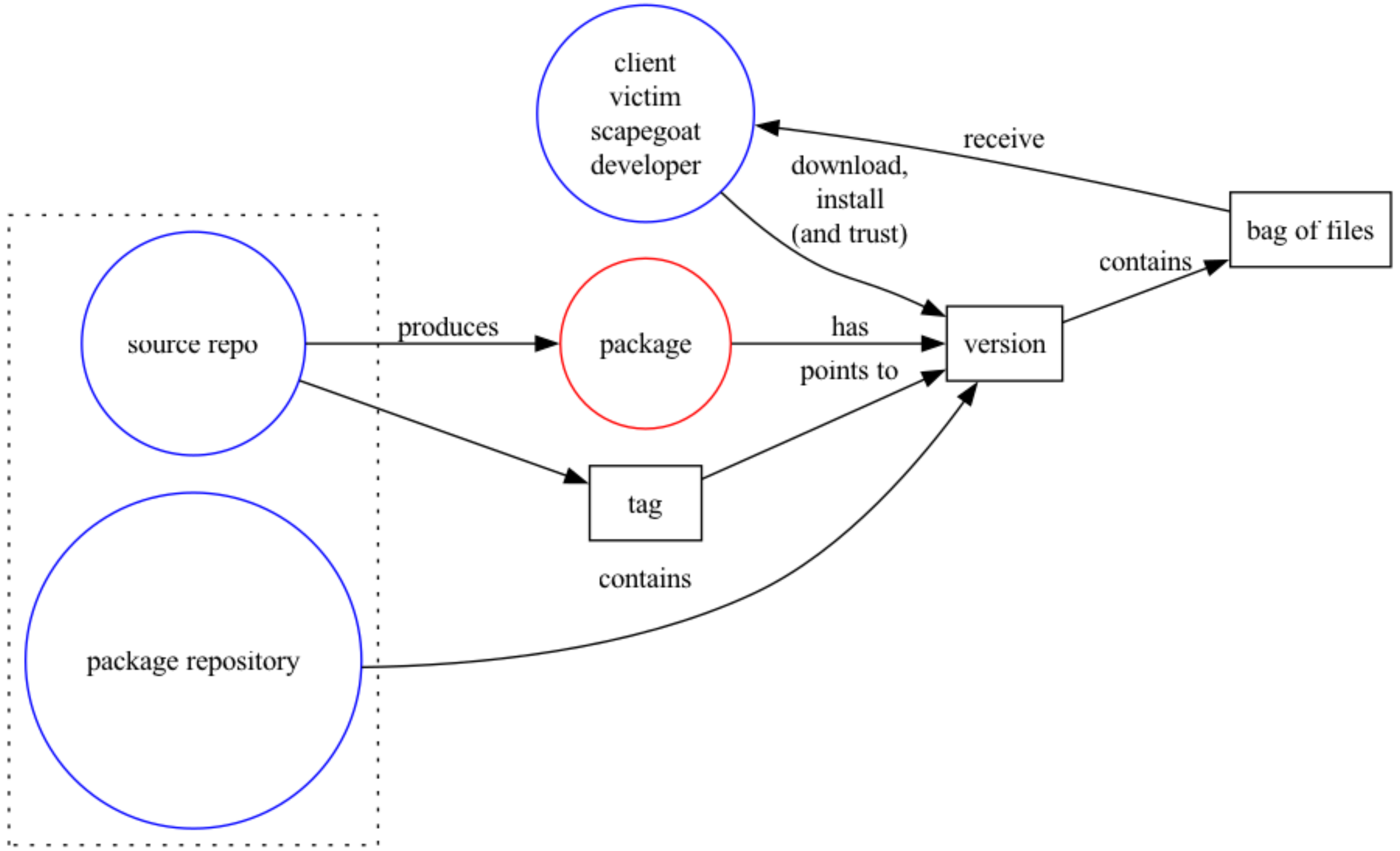
Web of trust

Each package has multiple versions



Web of trust

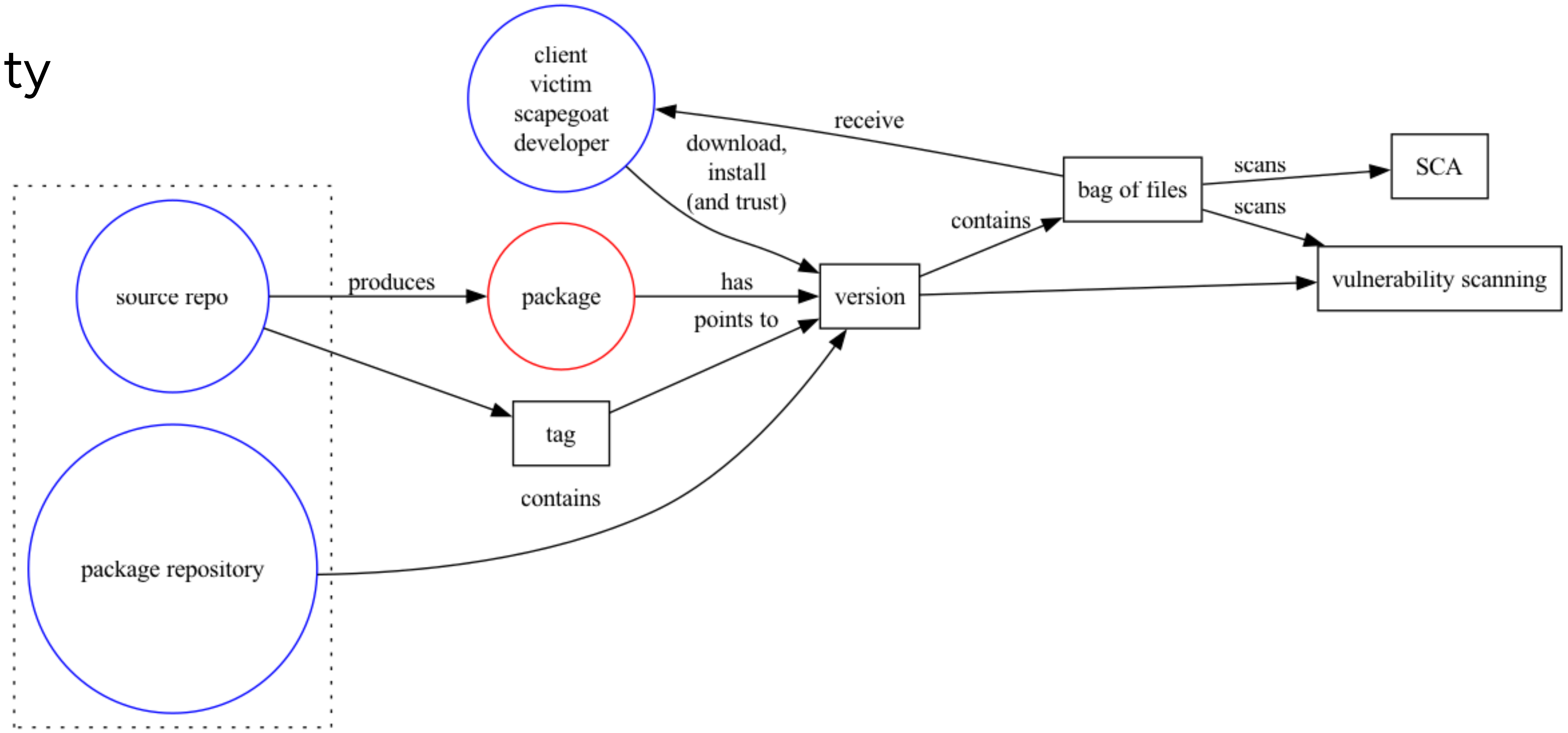
Normally delivered as a bag of files



Web of trust

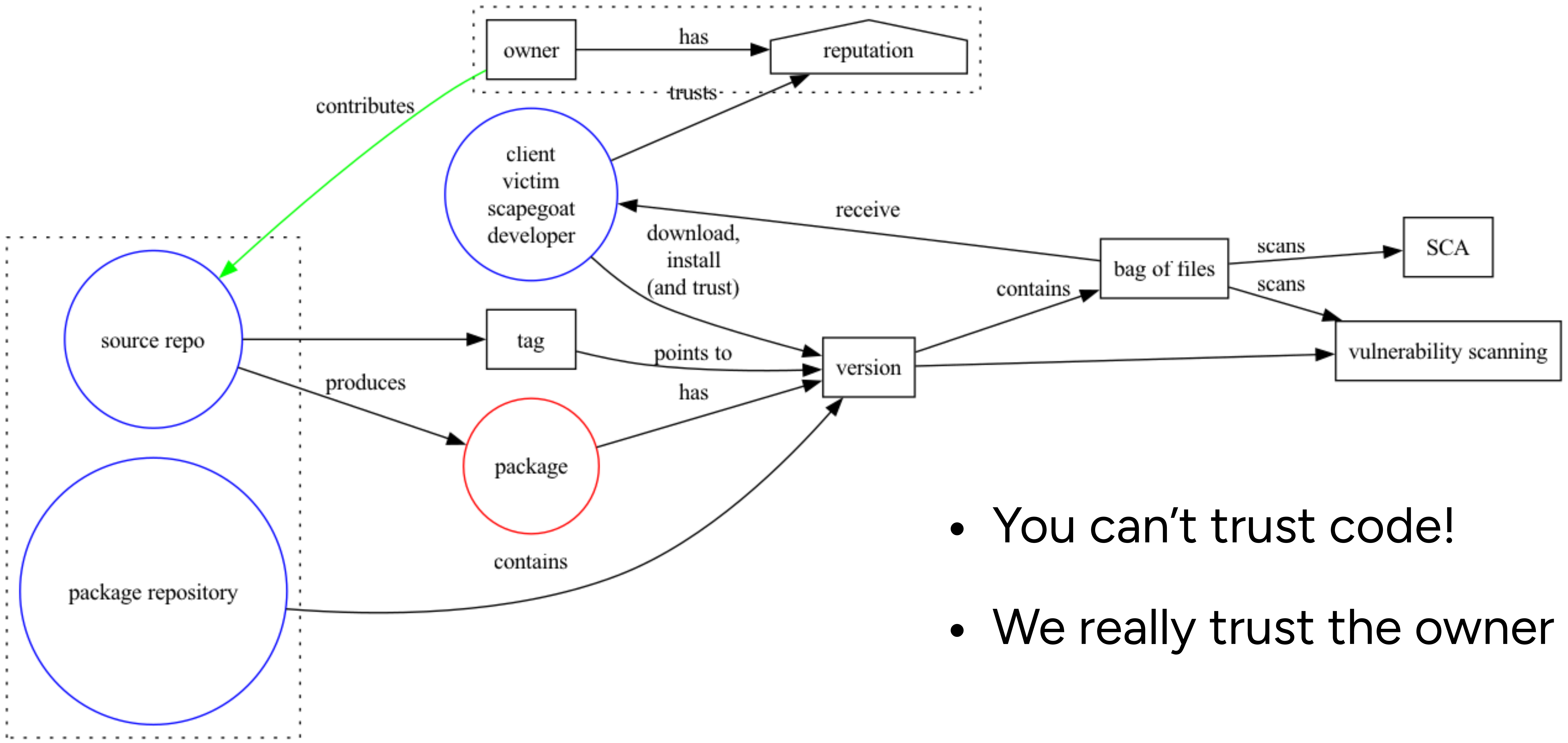
We want to scan it - see if it is good!

- Vulnerability
- SCA



Web of trust

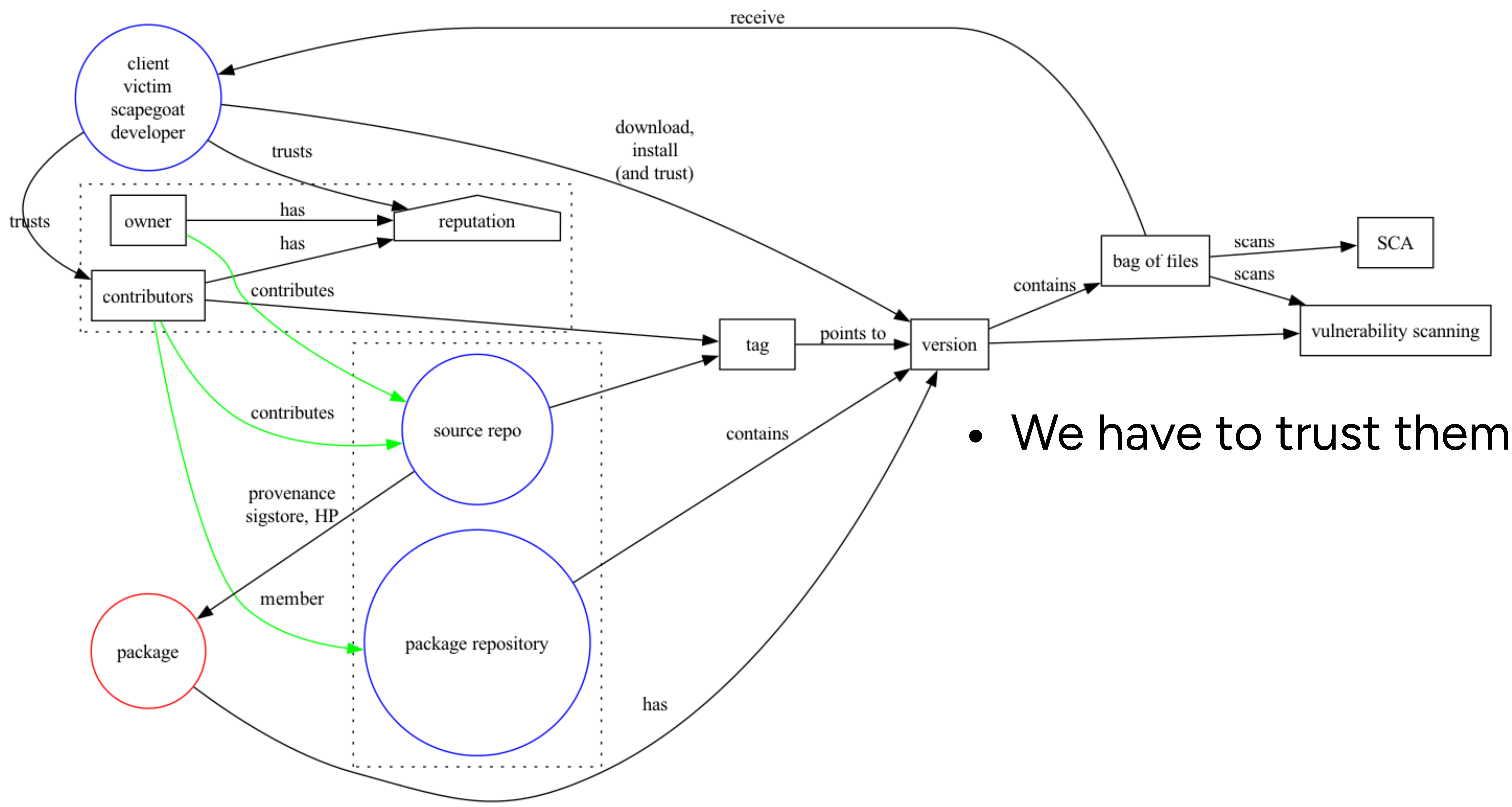
The code has an owner



- You can't trust code!
- We really trust the owner

Web of trust

There are other contributors

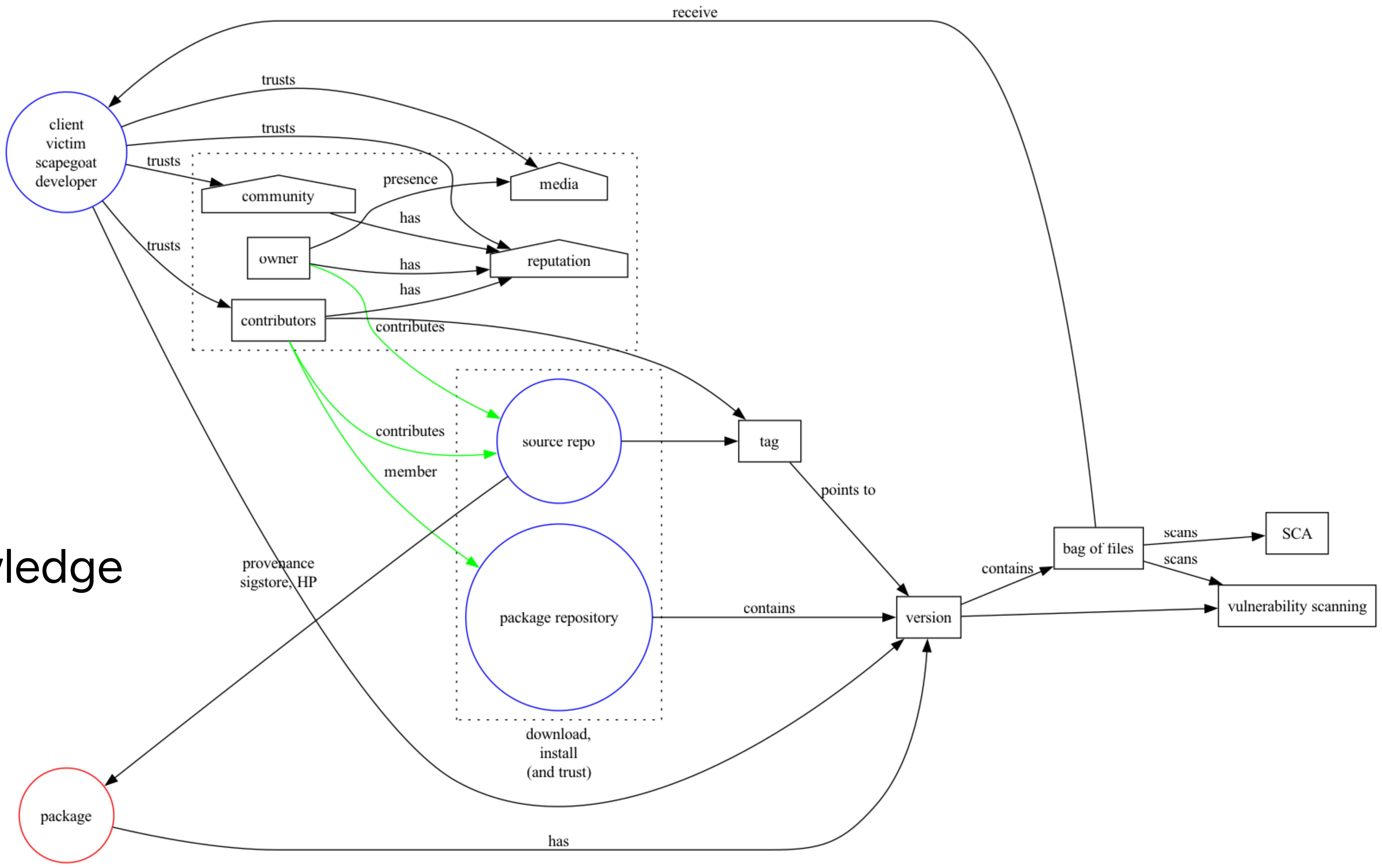


- We have to trust them too!

Web of trust

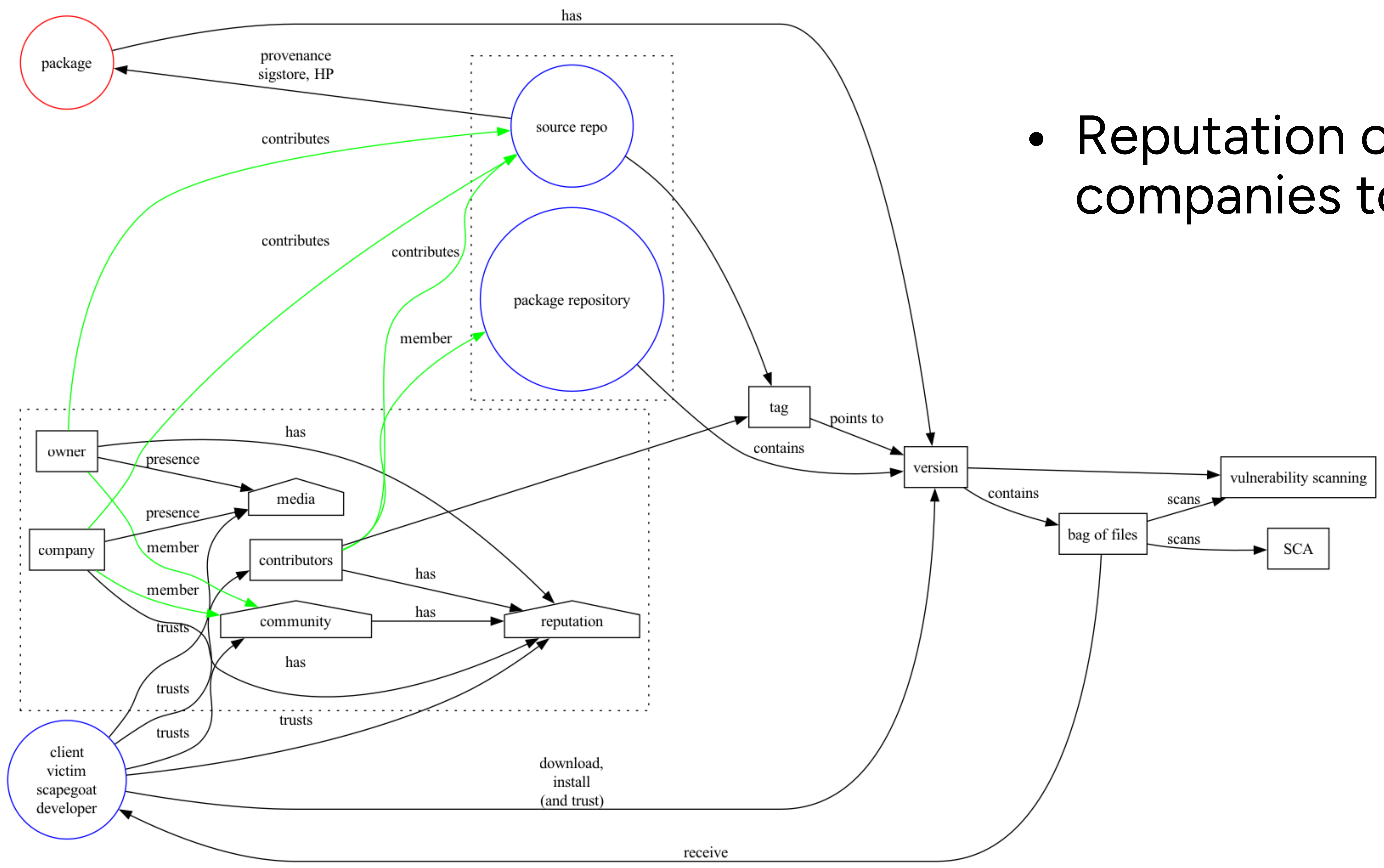
Communities

- Reputation from
  - media
  - personal knowledge
  - community



Web of trust

Companies are involved



- Reputation can come from companies too!

Transitive dependencies

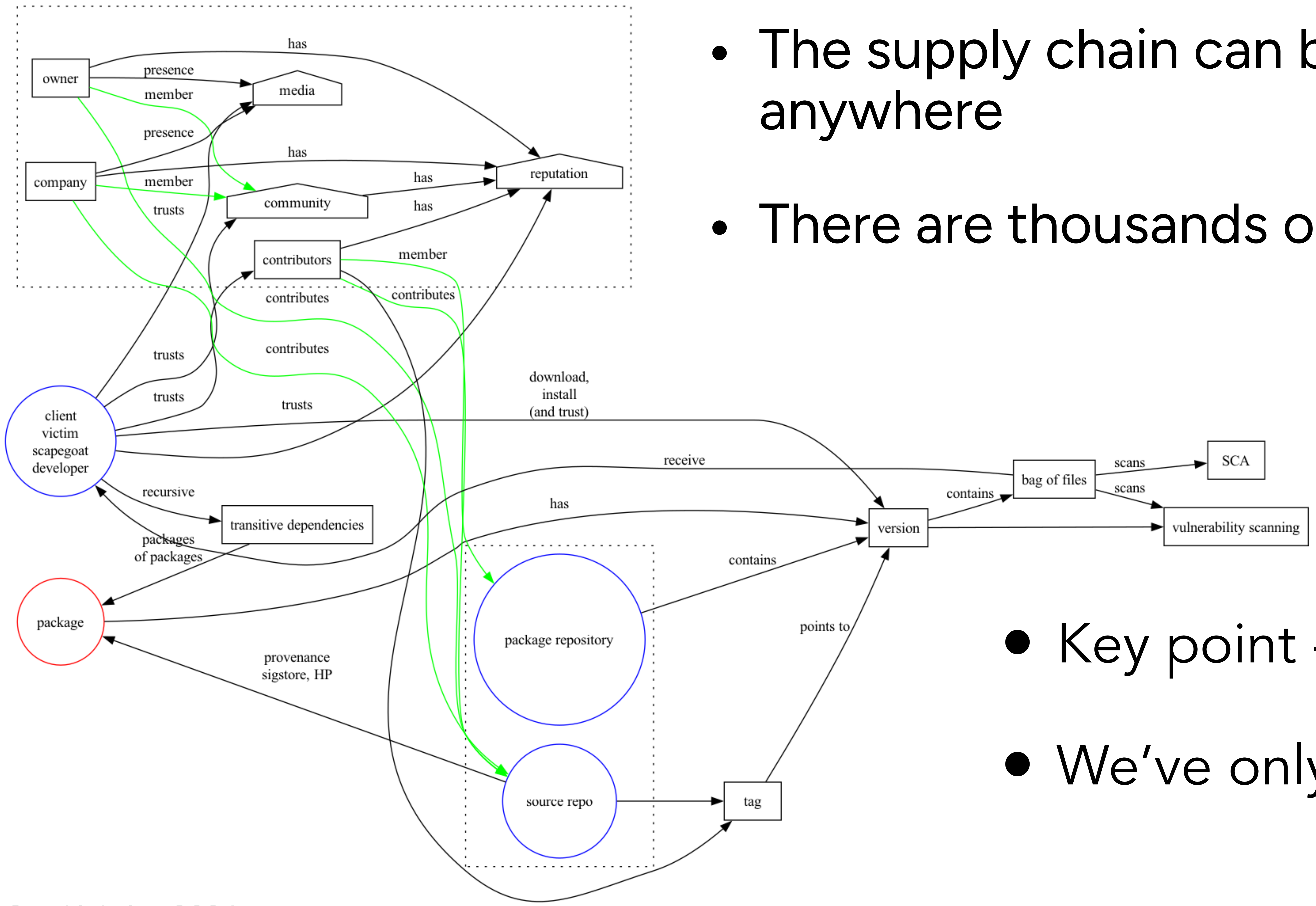
Or, Turtles all the way down

- We have packages of packages (Ave 1500 deep)
- You could investigate one package manually
- Thousands, you can't
- Key point -
  We need automation

Image: https://www.testifysec.com/blog/turtles-all-the-way-down/

Web of trust

Complex and fragile!



- The supply chain can be attacked (or break) anywhere
- There are thousands of ways to draw this!

- Key point - this has complexity
- We've only just started

What to do?
Currently

- CVEs
    - We can count them
    - And fix them
- Static Code Analysis
    - Mostly signature based
    - We'll do more of this
        - 3rd party and our own
- We should definitely CVE and SCA
    - But that's a story for another day

Meta-data

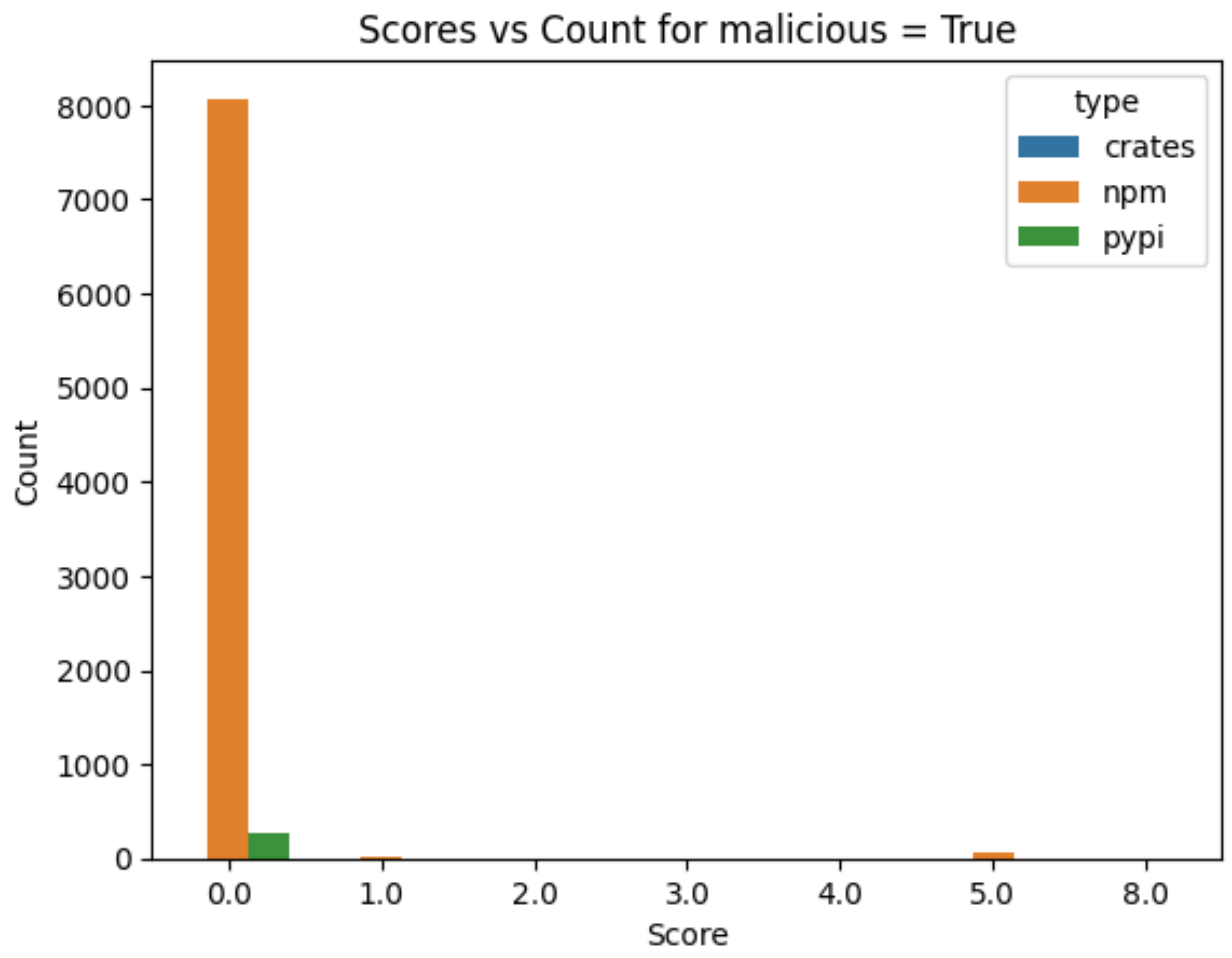
Is malice apparent?

- Malicious packages look different
  - Some look fine
    - Most malicious packages make no effort
      - Many legit packages make no effort

Looking for malice in meta-data

The bad apples

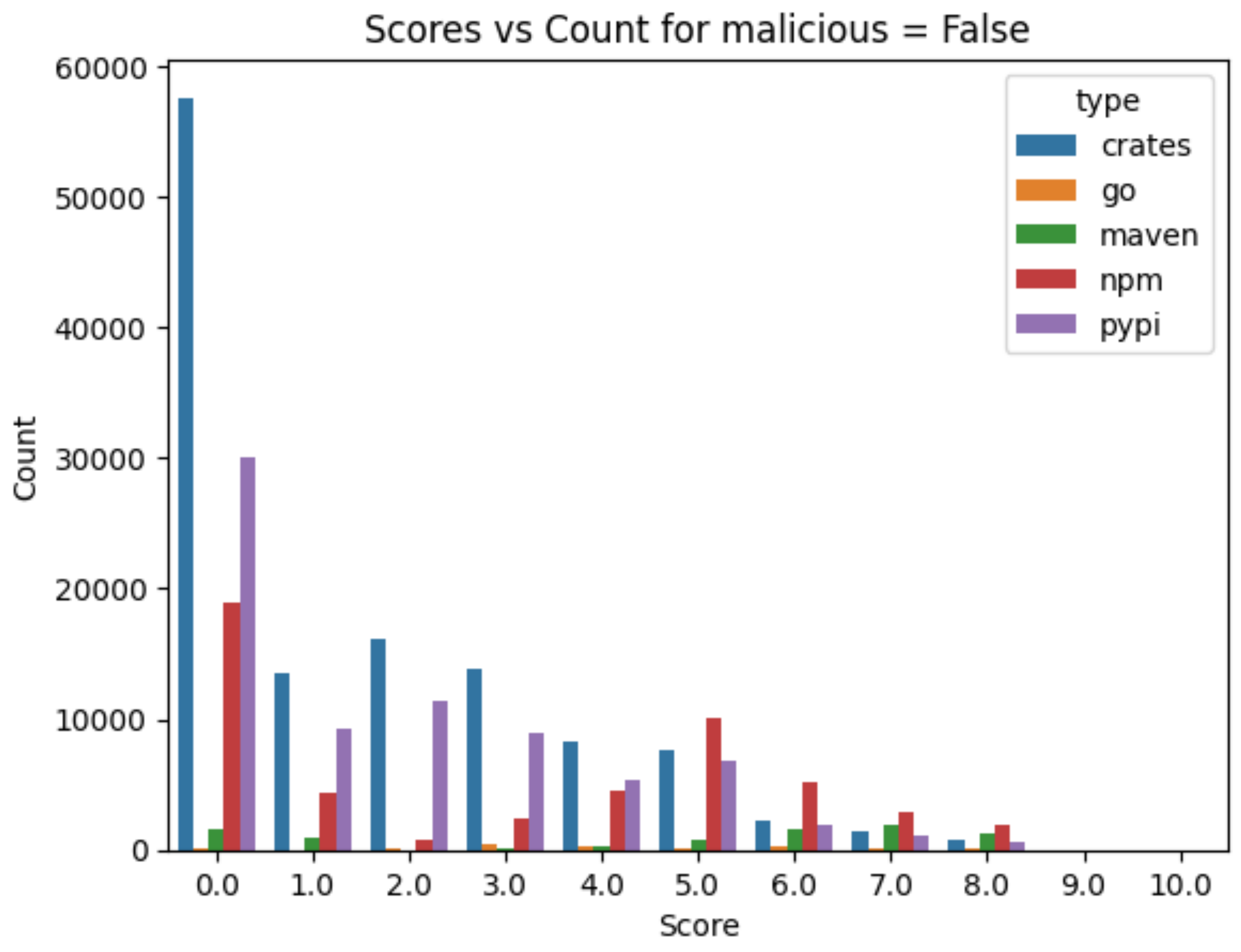
- Create a score based on
    - Activity
    - Provenance
    - Normalise it
- Compare
- Looks like we can spot malicious files!



Looking for malice in meta-data

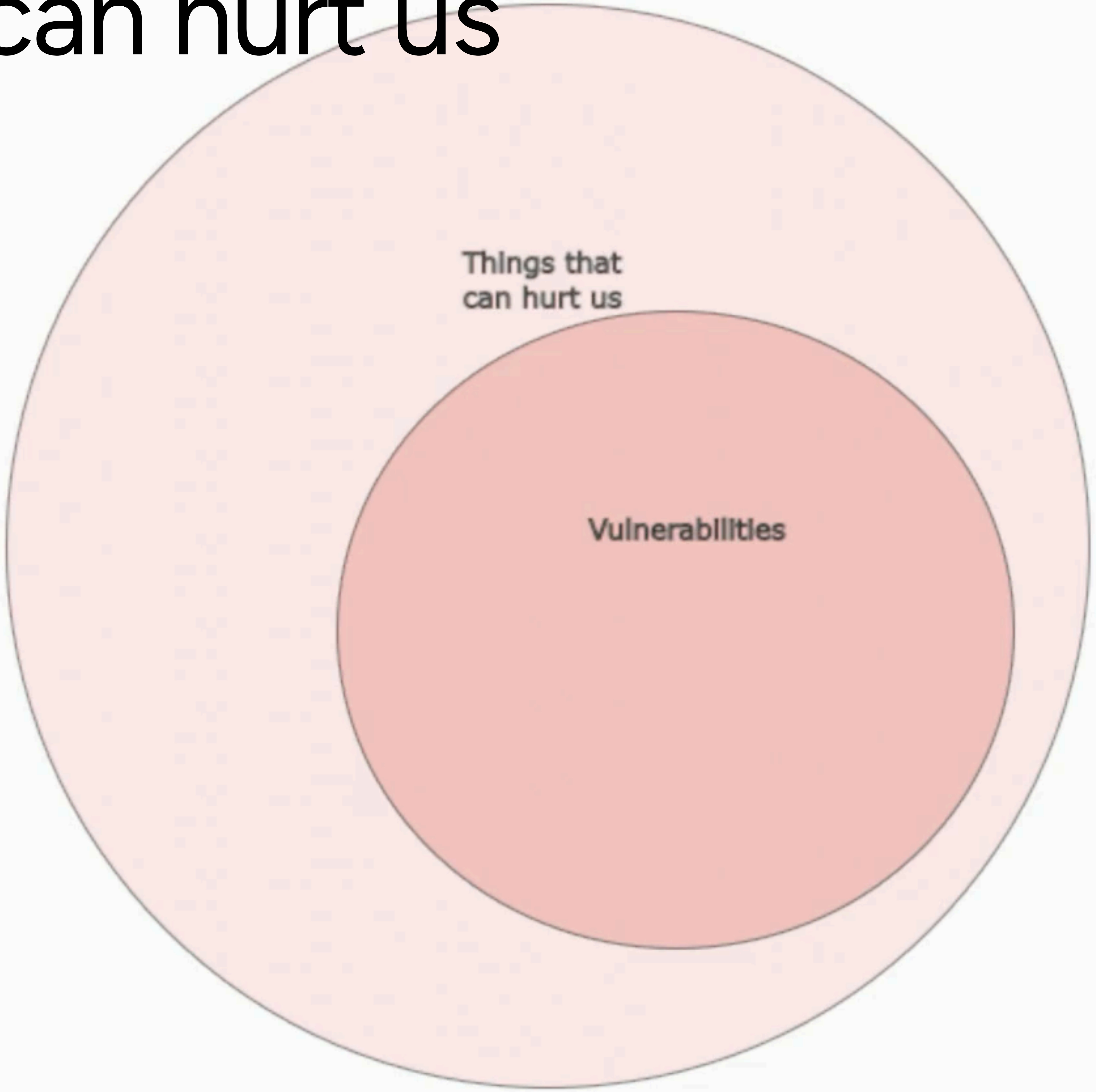
All the apples

- 10x Non malicious files score low
- If we get a low score, 1/10 chance it is malicious
- Looks like we can't spot malicious files!
- Does this matter?
    - Probably not
    - Many favourites score high



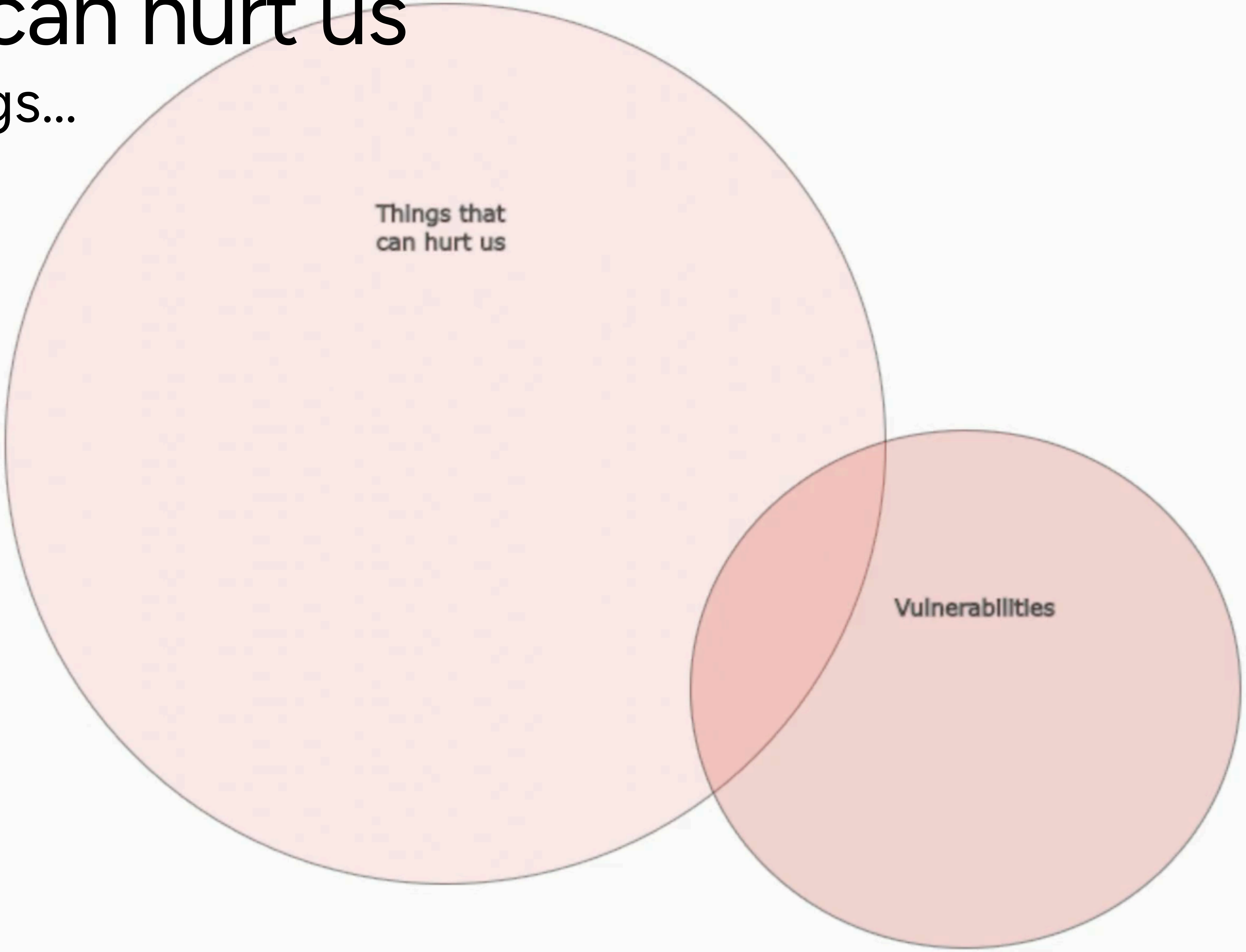
Things that can hurt us

We act like this...
Its not like



Things that can hurt us

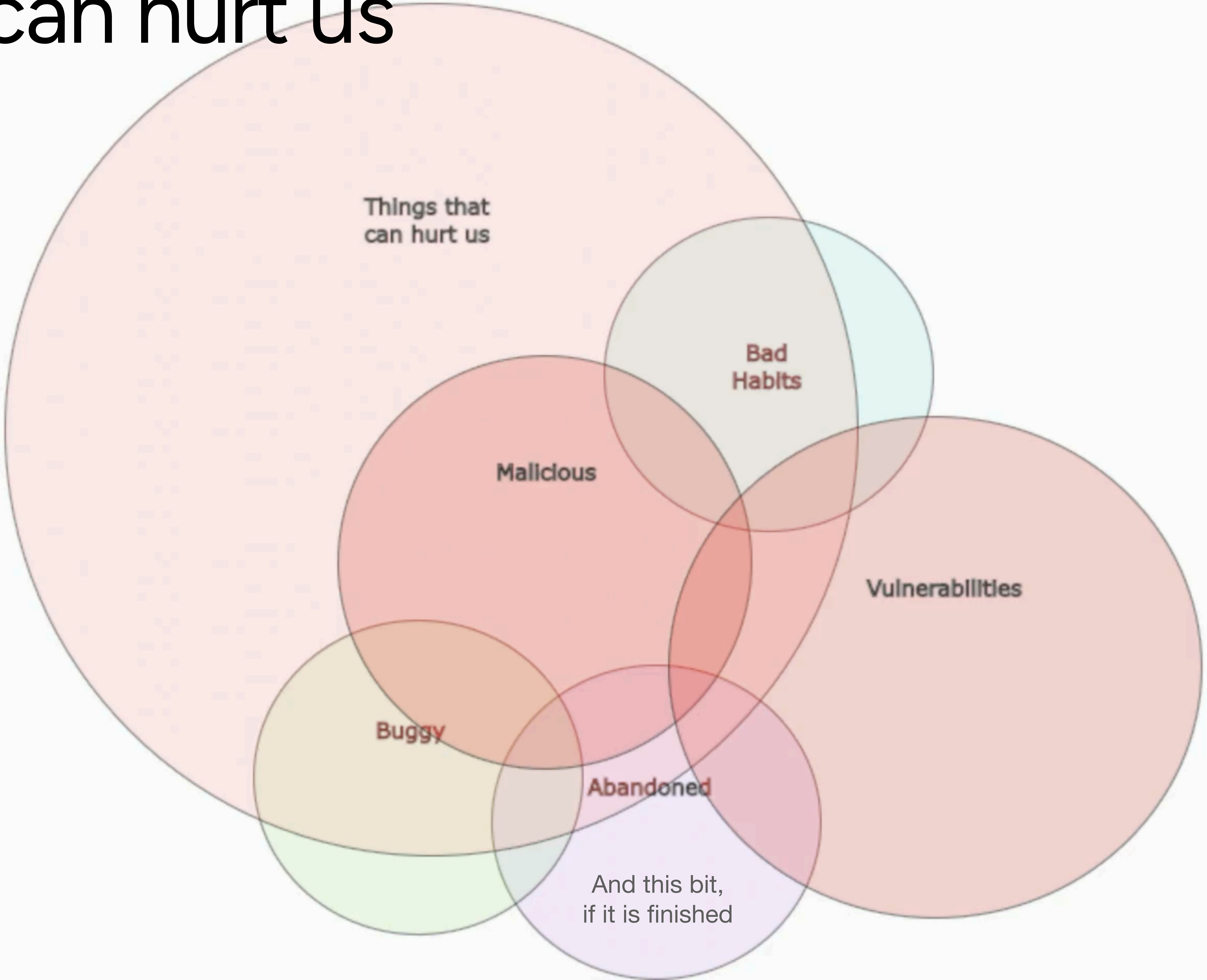
Other hurtful things...



Things that can hurt us

Most of it hidden

We want this bit.



And this bit,
if it is finished

Things that *probably* won't hurt us

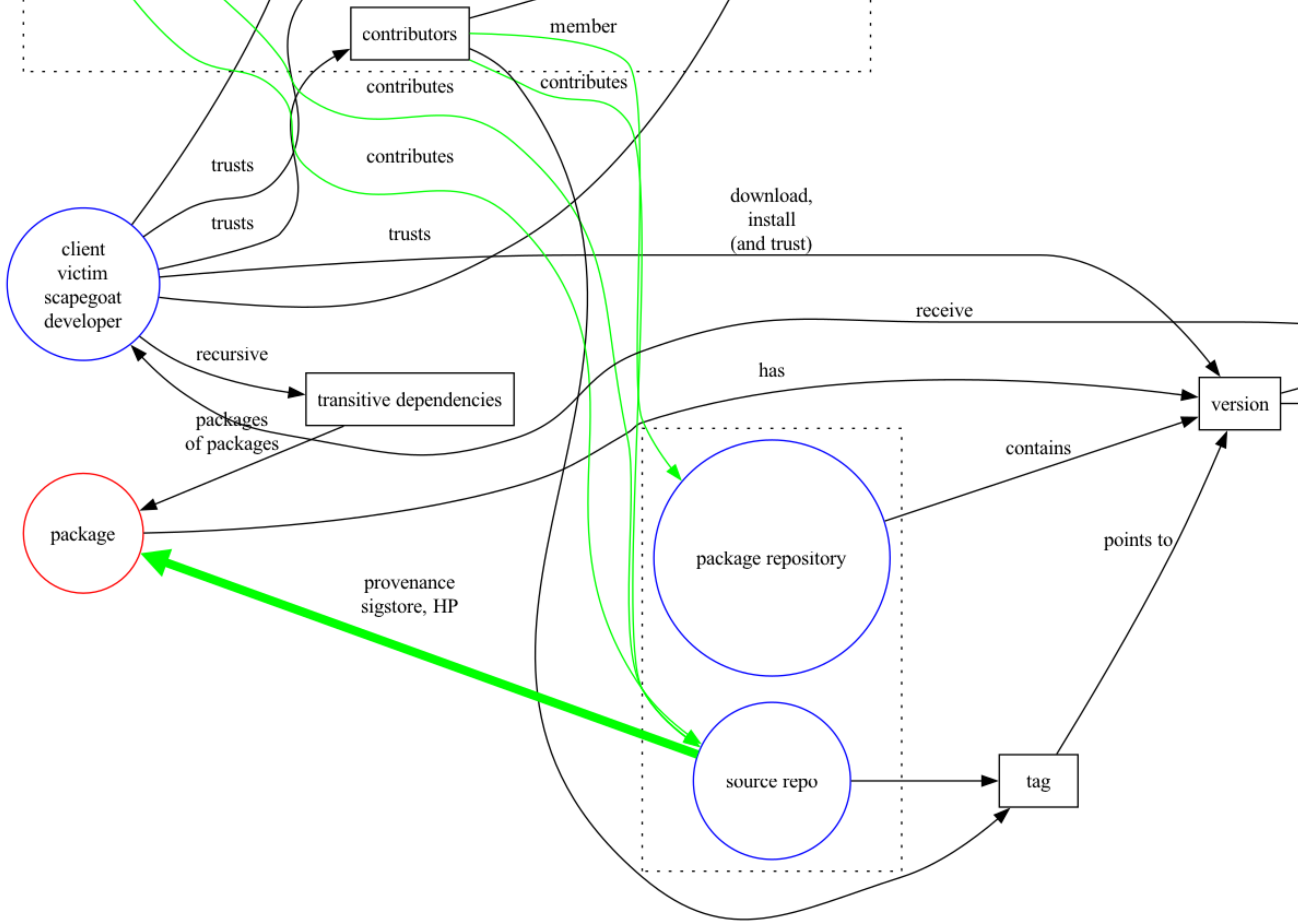
Look for the good apples

- Good habits/code hygiene
- Active development
- Developers we trust
- CVE and SCA clear
- Key point
    - Looking for good things is easier because it isn't hidden

Provenance

Is it what is says it is?

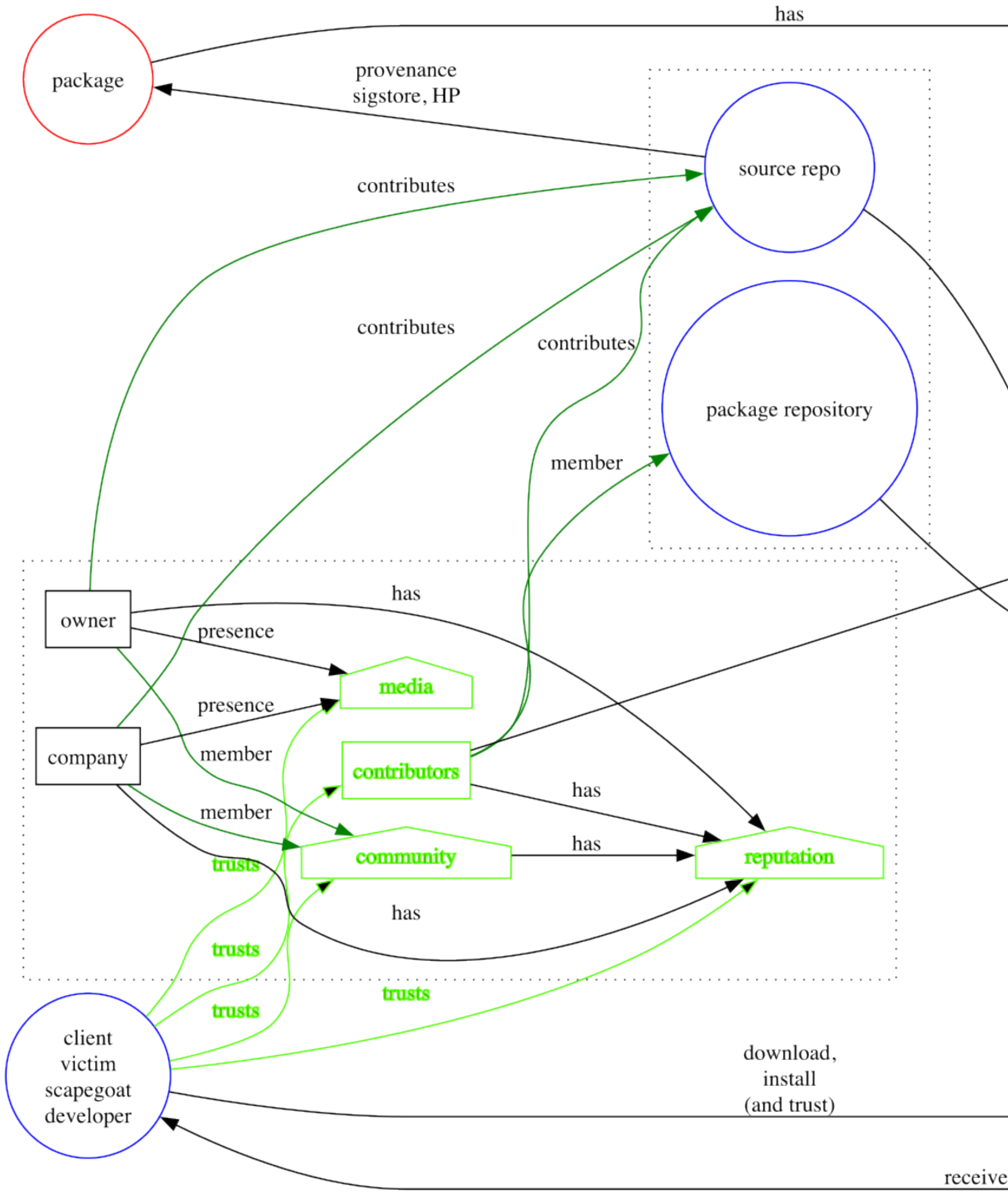
- SBOMs
- Sigstore
- Historical Provenance



Is code any good?

No short cuts

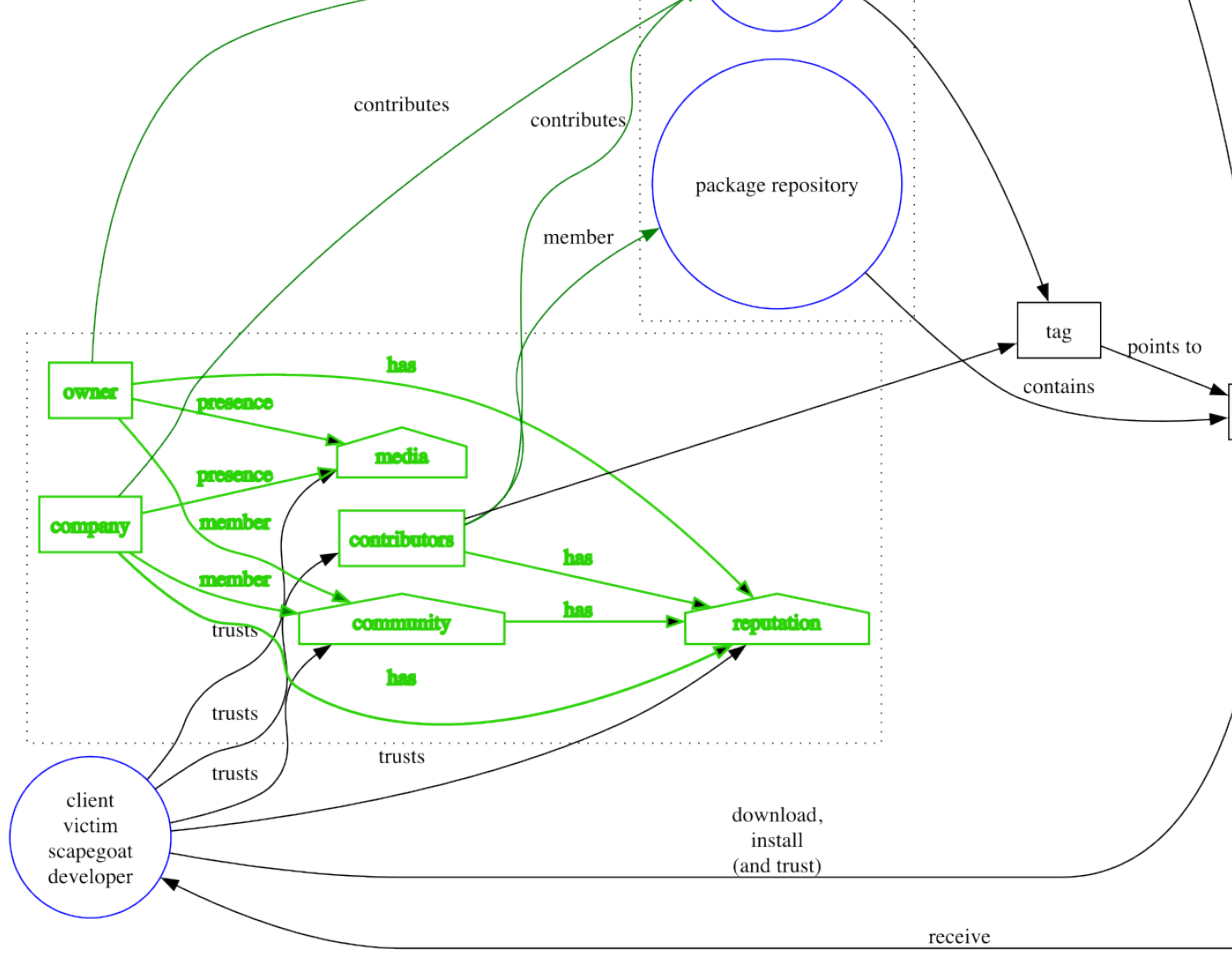
- Test it
- Measure it SCA
- Code review (requires provenance)
- Become intimate with it…
- Key point -
  Share the work with a community!
- Automate this



Reputation

Where does it come from?

- We know someone
- We know a company
    - They're big
- We guess…
- Hope…
- Do we even care?
    - (Yes! EO says so…)

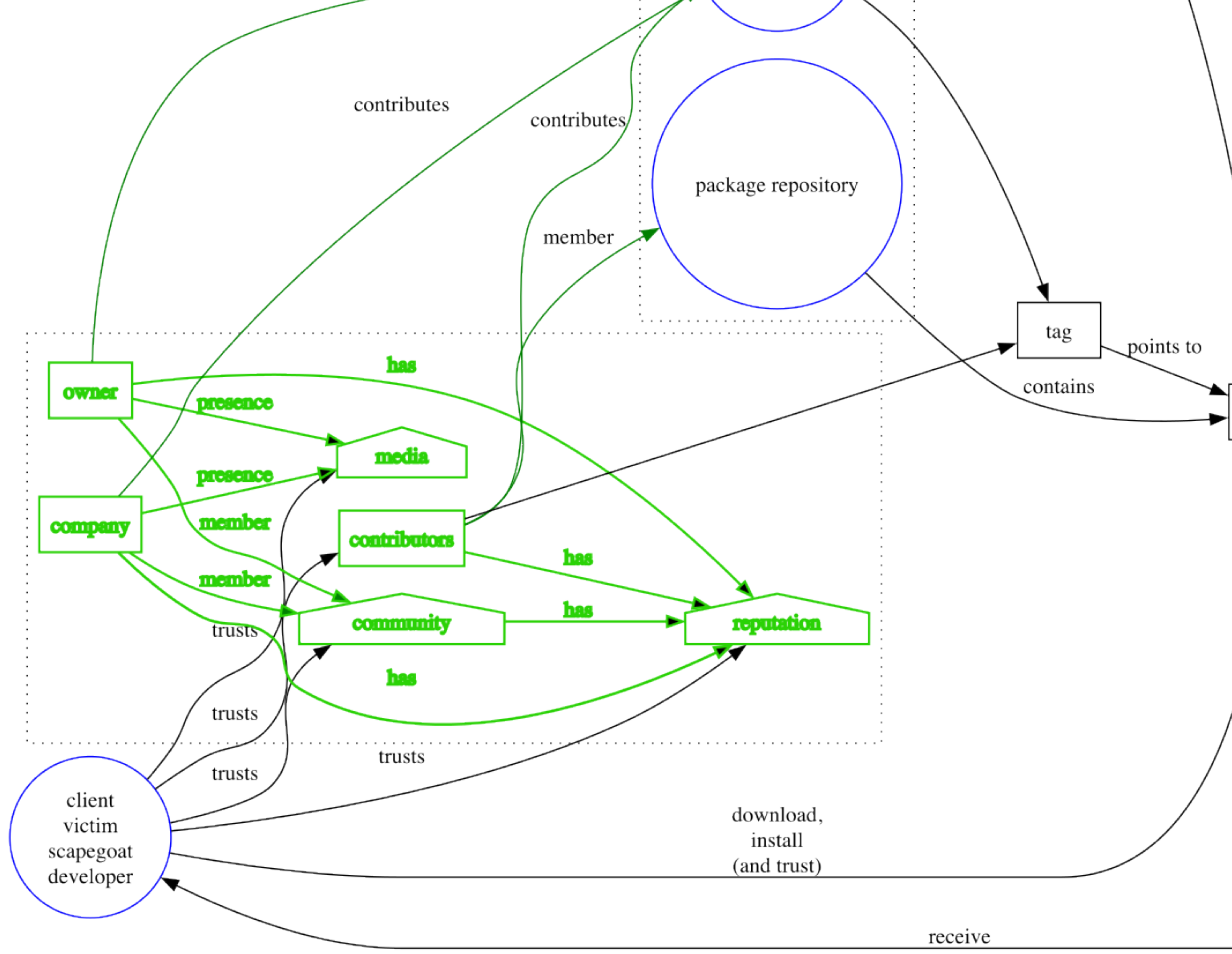


Reputation

Where should it come from?

- We should look at
  - participation
  - prior art
  - recommendations
- Generally, proof!

- Again, automate!



Key points

tl;dr

- Look for good things - easier to spot
- You don’t trust code, you trust people
- Trust is complex - it can break in many places
- Reputation is important
- Communities can share work
- Automation makes this possible at scale

stacklok

- Check out what we are doing
  - https://stacklok.com/
  - Discord
  - https://www.trustypkg.dev/

Questions?

Key points

tl;dr

- Look for good things - easier to spot
- You don’t trust code, you trust people
- Trust is complex - it can break in many places
- Reputation is important
- Communities can share work
- Automation makes this possible at scale